Difficulties to enforce your privacy preferences on Android? Kapuer will help you

Arnaud Oglaza
Toulouse France
Email: contact@kapuer.org

Romain Laborde
University of Toulouse
IRIT UMR 5505
Toulouse France
Email: laborde@irit.fr

Pascale Zaraté
University of Toulouse
IRIT UMR 5505
Toulouse France
Email: zarate@irit.fr

Abdelmalek Benzekri
University of Toulouse
IRIT UMR 5505
Toulouse France
Email: benzekri@irit.fr

François Barrère
University of Toulouse
IRIT UMR 5505
Toulouse France
Email: barrere@irit.fr

***Abstract*—Smartphones and mobile computing have changed our world and we are now over connected. Millions of applications are available to help us in every way possible. However applications can collect data from users for different purposes. Many private data are used to profile users. How to control privacy in this environment ? We propose a system called Kapuer that improves the management of applications permissions on Android by combining access control and decision support. We present in this article the Android implementation of Kapuer.**

I. Introduction

Within recent years, our environment has been transformed and we are now flooded in electronic devices and computers. The evolution has been so strong that our watches, phones and even glasses are now powerful computers. Android[1] is actually the most used mobile operating system with more than one billion new units sold in 2014 and around 1.4 million applications available on the Google Play Store (the official marketplace for Android applications). Android applications developers can access hardware features of devices like GPS location or the camera but also to users' data such as contact lists or calendars. To actually have access to each feature, the developer has to ask the associated permission. When someone installs an application, Android informs him about the list of permissions this application requires. At this time, the user has to make a simple choice: accept to give the application unlimited access to all the requested permissions or decline and cancel the installation. Furthermore, it is not always clear why an application requires some permissions. No explanation is given to the user nor information about when the application will access the resource nor the purpose. These practices can lead to severe issues for users privacy. Some customs Android release provides tools to manage permissions. CyanogenMod[2] is one of these customs OS and it includes its own permission management feature called Privacy Guard Manager (PGM). With PGM, it is possible to enable or disable each permission for each application. As number of applications installed on smartphones grows, this approach of managing permissions will face scalability issue. Access control models propose abstractions to group entities. For example, RBAC (Role Based Access Control) gives roles to users and create rules based on those roles. Why not applying abstractions to Android permissions management and create high level rules reducing the number of rules needed in the policy. Access control systems providing such feature give users a way to control the disclosure of their private data but suffer accessibility or efficiency to enforce their authorization policies. We have developed Kapuer, a generic and user-friendly approach based on a recommender system that learns users preferences in terms of privacy to help them control access to their private data. In this article, we present the Android implementation of Kapuer which helps users manage permissions given to applications installed on an Android device.

[1]http://www.android.com/
[2]http://www.cyanogenmod.org/

II. The Android Implementation

We developed a first proof of concept of a Kapuer integration on Android [1]. In this article, we present a full implementation of Kapuer for Android. This version is freely available to download at the following address : http://www.kapuer.org. We are making an open access beta test for further evaluations. The test started on the end of April 2015 and we have already around 350 downloads of Kapuer.

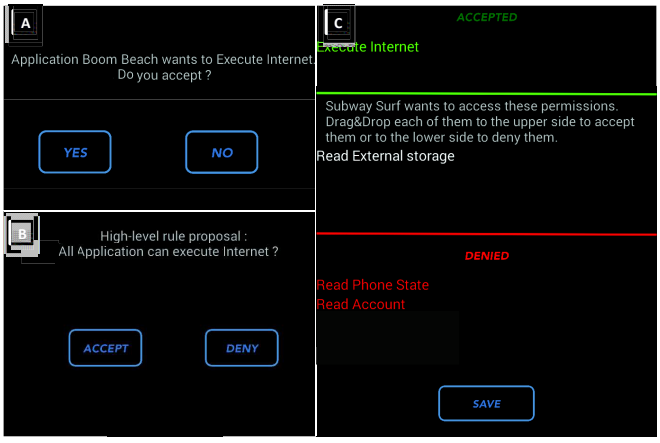

Fig. 1. Preferences learning and authorization rule proposal

We have chosen to work with three elements: the application making the request, the resource requested and the action to make on the resource. The two latter are derived from the permission requested. Then, for each element, we have multiple level of abstractions used to minimize the number of rules in the policy. To learn user's preferences, Kapuer interacts with him each time a permission request is made by an application and no rule exist to handle this request. Figure 1 shows all the possible interactions with the user. The two on

the left are the basic interactions; the one where Kapuer has no rule to handle the request and asks the user to make a decision (figure 1A) and the one where Kapuer has enough information on the user's preferences and proposes a high level rule (figure 1B). The third one (figure 1C) has proved to be necessary with the first real test of Kapuer. Indeed, some applications asks multiple permissions at their launch and it can quickly be annoying for the user. To avoid this situation, we have created an extension of the first interaction. This new one appears if an application requests at least a second permission just after the first. It asks a decision from the user for all remaining permissions of this application. This way, Kapuer avoid to disturb the user five or six times. Accepting a rule is not permanent, a rule can be deleted in the rule management section if the user mis-clicked during an interaction or just if he wants to change his privacy policy.

Kapuer offers an interface to inform the user about his privacy policy and tools to manage it. Figure 2 shows three screenshots of the application. All the rules of the policy are displayed in the rule list (figure 2A), by clicking on one rule, the user access to the screen of rule management (figure 2B). Here he can modify the rule on four parts: the application, the resource, the action or the decision. For the first three elements, he can either let the rule as it is or modify the level of abstraction of each parts. For instance, in the figure the application used in the rule is Dropbox and its category is Productivity (the category are the one found in the Google Play Store). Then the user can let the rule only apply for Dropbox or change it to apply for all Productivity applications. If he chooses the latter, an explanation text informs him and shows all the Productivity applications. Figure 2C displays the permissions list of an application and gives useful information to the user like permissions that can constitute a risk for his privacy and permissions handled by one of Kapuer's rule.

III. Kapuer VS Privacy Guard Manager

We wanted to test the approach with a real life situation. We have used the 50 most downloaded free applications in the Google Play Store and listed the permissions for each application to make a list of possible requests. Then we have created an eight rules long privacy policy to be applied on our test Android device. We want to evaluate the cost of writing such policy using Kapuer and Privacy Guard Manager (PGM) of CyanogenMod. This cost is the number of actions the user has to perform. For PGM, an action consists in all pressures (screen navigation and on/off flipswitches selection). For Kapuer, any interaction as explained in figure 1 is an action. Since the Kapuer learning process is not predetermined and depends on the received requests, a large number of tests must be executed to get its average behavior. Thus we have used our simulator presented in previous articles [2], [3] that can automate this task. It allowed us to run simulations with the same objective but with different request each time.

To create the whole policy, when PGM requires 848 actions, Kapuer only needs 190 actions in average. Thanks to the use of abstractions in the learning phase of Kapuer. Some rules proposed by Kapuer can handle tens of permissions where PGM needs one rule for each.

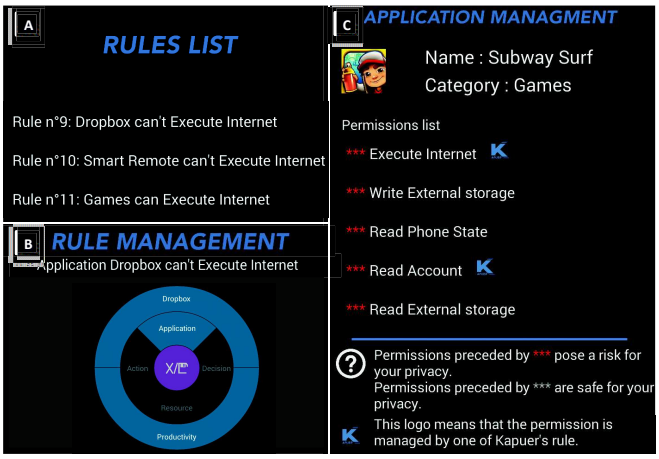

Fig. 2. Rule management interface

IV. Conclusion and Future Work

We have presented in this article a tool for permission management on Android. Unlike the other approaches, Kapuer doesn't only provide a way to modify what permissions an application can use. It learns from the user's behavior to help and advise him by proposing rules with different level of abstraction. This way, users can protect their privacy more easily, without needing knowledge about access control models or policy's structure. Evaluation shows that hundreds of permissions can be handled with few actions by using abstractions

One of the initial goal when we designed Kapuer was to inform people about privacy risks. Now we want to go further in that direction and not only inform people but also educate them about privacy issues. As an example, explaining them the consequences of granting some permissions to an application. The more people understand these risks, the better their privacy decisions will be.

Today Kapuer learns users preferences from scratch. A large number of requests is needed before any proposition can be made to the user. It is possible to improve the beginning of the learning phase by initializing the system. We are currently making surveys with different kind of users to find the best way to initialize users preferences.

References

[1] A. Oglaza, R. Laborde, and P. Zarate, "Authorization policies: Using decision support system for context-aware protection of user's private data," in *Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on*, July 2013, pp. 1639–1644.

[2] A. Oglaza, P. Zaraté, and R. Laborde, "KAPUER: A Decision Support System for Protecting Privacy (regular paper)," in *Group Decision and Negotiation (GDN), Toulouse, France, 10/06/2014-13/06/2014*, ser. LNBIP, P. Zaraté, G. Kersten, and J. Hernandez, Eds., no. 180. http://www.springerlink.com: Springer, juin 2014, pp. 100–107. [Online]. Available: http://oatao.univ-toulouse.fr/13069/

[3] A. Oglaza, R. Laborde, and P. Zarate, "Demonstration of kapuer: A privacy policy manager on android," *2016 13th IEEE Annual Consumer Communications and Networking Conference (CCNC) Demonstrations paper*, 2016 (accepted).